TEXT → SQL

1. System Overview

- What is RAG for Text-to-SQL?
A Retrieval-Augmented Generation (RAG) system that combines:

- **Retrieval**: Finding relevant SQL examples, schema information, and patterns from your knowledge base
- **Generation**: Using an LLM to create SQL queries based on retrieved context and user input.

2. Workflow

RAG for Text-to-SQL Process

3. Key Features

- **Schema Awareness**: Understands database schema (tables, columns, relationships) to generate contextually accurate queries.
- **Few-Shot Learning**: Leverages example question-SQL pairs for better performance.
- **Error Handling**: Validates SQL syntax and provides feedback for invalid queries.
- **Scalability**: Works with diverse databases by dynamically retrieving schema.

4. Setup and Requirements

- **Programming Language** - Python 3.8+
- **Libraries** - Transformers, Langchain, LlamaIndex
- **Vector Store -** ChromaDB, Pinecore, FAISS

5. Model Selection

AI and Embedding Model Comparison

Characteristic	AI Model	Embedding Model
OpenAI	GPT-4o, GPT-3.5-Turbo	text-embedding-ada-002, text-embedding-3-small
Gemini	Gemini-1.5-Pro, Gemini-1.5-Flash	text-embedding-004
Hugging Face	T5, CodeBERT, BART	HuggingFace Embeddings
Ollama	SQLCoder	all-MiniLM-L6-v2
Mistral	Mistral 4	Sentence-BERT (paraphrase-MiniLM-L6-v2)
Grok	Grok 3	Sentence-BERT (all-MiniLM-L6-v2)

6. Pricing Comparison :

Model	Provider	Input Tokens ($/1M)	Output Tokens ($/1M)	Embedding Tokens ($/1M)
GPT-4o	OpenAI	$5.00	$15.00	$0.13 (text-embedding-3-large) $0.02 (text-embedding-3-small)
GPT-4o Mini	OpenAI	$0.15	$0.60	$0.13 (text-embedding-3-large) $0.02 (text-embedding-3-small)
Gemini 1.5 Pro	Google	$3.50	$10.50	$0.10 (text-embedding-004)
Gemini 1.5 Flash	Google	$0.35	$1.05	$0.10 (text-embedding-004)
Llama 3.1 (via Ollama)	Ollama	~$0 (local hosting)	~$0 (local hosting)	~$0 (local, e.g., nomic-embed-text)
Mistral (via Ollama)	Ollama	~$0 (local hosting)	~$0 (local hosting)	~$0 (local, e.g., nomic-embed-text)

7. SQL Examples Repository :

Example :
json

```
{
  "id": "sql_001",
  "natural_language": "Get all employees with salary greater than 50000",
  "sql_query": "SELECT * FROM employees WHERE salary > 50000;",
  "category": "filtering",
  "complexity": "basic",
  "tables_used": ["employees"],
  "operations": ["SELECT", "WHERE"],
  "business_context": "HR queries for salary analysis",
  "tags": ["salary", "employees", "filtering"]
}
```

8. Sample Prompt

Prompt :
"""
System Prompt:
You are an expert SQL query generator. Generate accurate SQL queries based on natural language input and provided context.

Context Section:
Database Schema: {schema_information}
Relevant Examples: {retrieved_examples}
Query Patterns: {relevant_patterns}

Task Section:
Natural Language Query: {user_query}
Requirements: {specific_requirements}

Output Format:
Provide only the SQL query without explanations unless requested.

If you dont find any relevent query from vectore store, use ai model and extranal infromation for question.

"""

Task Distribution

1.Planning
Description: Define the project scope, objectives, and technical requirements.
2.Model Selection and Pricing

Description: Deep Rsearch on differnt AI and their embeddings models.

3.Documentation
Description: Create comprehensive documentation for the RAG model, covering setup, usage, and maintenance.

4.Data Collection
Description: Gather and prepare data for training, retrieval, and testing the RAG model.

5.Model Building
Description: model (Retriver and Generator) building for RAG.

6. Endpoint (If needed)
Description: Create a web UI to enter user input (Query).